Adaptive-Price Combinatorial Auctions

SÉBASTIEN LAHAIE, Google Research†
BENJAMIN LUBIN, Boston University

This work introduces a novel iterative combinatorial auction that aims to achieve both high efficiency and fast convergence across a wide range of valuation domains. We design the first fully adaptive-price combinatorial auction that gradually extends price expressivity as the rounds progress. We implement our auction design using polynomial prices and show how to detect when the current price structure is insufficient to clear the market, and how to correctly expand the polynomial structure to guarantee progress. An experimental evaluation confirms that our auction is competitive with bundle-price auctions in regimes where these excel, namely multi-minded valuations, but also performs well in regimes favorable to linear prices, such as valuations with pairwise synergy.

† Authors listed in alphabetical order. Contact at *slahaie@google.com* and *blubin@bu.edu* respectively.

1 INTRODUCTION

A combinatorial auction (CA) allows bidders bid on bundles of items, enabling them to express complement or substitute preferences, ideally leading to an increase in allocative efficiency. CAs have found widespread applications in domains ranging from spectrum license allocation [10], to TV advertising [15] and industrial procurement [30]. Because the process of valuing even a single bundle can be a costly exercise for a bidder [27], most fielded CAs include an iterative elicitation phase where prices are used to guide bidders in selecting which packages to bid on. A leading example is the Combinatorial Clock Auction (CCA), which has been used by government entities around the world to conduct spectrum auctions and has generated over $20 billion in revenue [3].

A key point of differentiation among the various iterative CA designs proposed to date is their choice of pricing scheme. Proponents of item-price auctions argue that their simple price structure can reduce the cognitive burden of package bidding and promote effective price discovery [19]. On the other hand, bundle-price auctions such as *i*Bundle [26] and the Ascending-Proxy auction [5] are supported by incentive analyses and provable efficiency guarantees. In practice, the CCA attempts to resolve this tension by using an ascending item-price clock phase together with a final round of sealed package bidding [3]. Even with the final round, the quality of the price feedback and final allocation in the clock phase remains crucial. In an ideal scenario, the clock phase would reach an efficient allocation (without over- or under-demand), and the final round would simply serve as a price adjustment to encourage truthful bidding. However, under the complex bidder valuations for which CAs are used, linear prices are usually too restricted to clear the market [8].

In this paper, we employ prices at a middle ground between linear and bundle prices, and use them to achieve informative price discovery and market clearing in tandem. We design the first iterative CA with fully *adaptive* pricing, which augments the price structure as the auction progresses. We implement our auction using *polynomial* prices. Polynomial prices are a natural extension of item prices where combinations of items may be priced as well; for instance, *quadratic* prices add a surcharge to pairs of items. This added expressivity allows polynomial prices to clear the market and reach efficient allocations for a wider class of valuations than linear prices, and with enough price terms they can reach full expressiveness like bundle prices. Our auction begins with linear prices, and introduces prices on combinations of items as needed. Our adaptive pricing scheme can provably detect when the current price structure is insufficient to clear the market, and correctly select a price expansion (i.e., which new price term to introduce) to guarantee progress.

We conduct an extensive simulation study and find that our adaptive polynomial-price CA can simultaneously achieve fast convergence and high efficiency on diverse classes of valuations. By contrast, simple item- or bundle-price auctions necessarily fail on some of the classes, either by requiring large numbers of rounds, or by failing to converge altogether. We use *i*Bundle for our baseline bundle price auction, which is closely related to the Ascending-Proxy auction [5, 26]. For our baseline linear price auction we use the clock phase of the CCA as described in Ausubel and Baranov [3]. We also compare our auction to a simplified linear price one, isolating the benefit of adaptive pricing. We find that even in domains where linear prices typically clear the market, adaptive pricing can substantially speed up convergence with virtually no impact on efficiency.

Our Techniques and Results. The design of our auction follows the LP approach to iterative auction design pioneered by de Vries et al. [13], building on the work of Bikhchandani and Ostroy [8]. Under this approach, the efficient allocation problem is formulated as a linear (rather than integer) program, and an auction corresponds to an algorithm for solving this LP. de Vries et al. [13] observed that all the iterative CAs in the literature (at least, those with provable efficiency guarantees) could be categorized as either subgradient or primal-dual algorithms to solve the allocation LP: the steps of the algorithms have natural auction interpretations as bidding, provisional allocation, and price update. To ensure that the auction converges to an efficient allocation, the crucial property is that the LP formulation should have an integer optimal solution.

Our auction is an instance of the subgradient class, based on its price update method, but our test for price expansion uses ideas from primal-dual algorithms. According to the LP approach, testing for price expansion is equivalent to testing whether the associated LP formulation of the allocation problem is integer. The key difficulty is that, during the auction, the bidder valuations that form the objective of the LP are not available, only the current bidder demands. Our main result (Theorem 3.2) shows that it is enough to test whether a *restricted primal* LP based purely on the demands is integer. This leads to a price expansion policy (Corollary 3.3) that tests whether 1) further progress is possible with the current price structure, 2) the auction has converged, or 3) price expansion is needed to clear the market. Although we prove these results in the context of polynomial prices for concreteness, they would straighforwardly extend to more general families of price structures (at the cost of more abstraction and notation). For polynomial prices in particular, our final result (Theorem 3.4) shows how to generate a new price term to guarantee progress, or confirm that personalized prices are needed to clear the market.

The second main contribution of this work is an extensive empirical evaluation of our adaptive-price auction against benchmarks including *i*Bundle and the clock phase of the CCA. We consider valuations generated from the standard CATS test suite [21] as well as valuations with very different structure (still motivated by real-world instances) generated according to the Quadratic Model proposed by de Vries and Vohra [14]. Our key finding is that the adaptive-price auction is the only auction able to perform at high levels of efficiency and convergence rate across all domains. The lowest average efficiency of the adaptive-price CA across domains is 94%, whereas for the benchmarks it drops to 37–81%. Our auction is also able to clear all instances, except for instances from a specific CATS distribution that turn out to require personalized prices (as detected by our price expansion scheme); even for these instances, efficiency is high (98%) upon termination. In terms of speed of convergence (measured in rounds), adaptive pricing again leads to the best average performance across valuation classes. Taken together these results suggest that, unless one has detailed a priori knowledge on the structure of bidder valuations, adaptive pricing is a method of choice to achieve high clearing, efficiency, and speed of convergence.

Practical Considerations and Incentives. The focus of this work is on price structure and price updating in iterative CAs. Of course, many other aspects of design are also important, including activity rules and reserve prices [3]. To induce bidders to place best-response bids and drive the auction forward, the literature offers several techniques. One approach is to maintain multiple price paths, effectively running several auctions in parallel (one with each agent removed) to eventually elicit enough information to compute VCG payments [23]. Alternatively, one could view our adaptive-price auction as a possible replacement for the clock phase of the CCA, and apply a suitable payment rule in a sealed-bid phase to induce good incentives. Payment rules for the CCA are an active area of research, including detailed evaluations of VCG and core-selecting payments [12, 20]. If our adaptive-price design were used in practice as part of a complete combinatorial auction, then one would also use activity rules and payment rules to induce good incentives. For this reason, we consider the incentive problem to be orthogonal to the price update problem. Thus, for the remainder of this paper, we follow prior work (e.g., [7, 26, 28]) and assume that bidders follow myopic best-response (straightforward) bidding throughout the auction.

Related work. Combinatorial auction formats with both linear and bundle prices have been studied extensively through theory, simulation studies, and lab experiments. The arguments in favor of bundle price auctions are that they lead to high levels of allocative efficiency in both theory and practice. Parkes [26] introduces *i*Bundle, a combinatorial auction with personalized bundle prices, along with a dynamic variant called *i*Bundle(d) that switches from anonymous to personalized prices when needed. Ausubel and Milgrom [5] introduce the Ascending Proxy auction, closely related to *i*Bundle, and focus on its incentive properties. Both auctions provably achieve high efficiency as long as agents follow straightforward bidding.

An important argument in favor of linear price auctions is that item prices provide informative feedback for the bidders, alleviating the difficult cognitive problem of package bidding, and leading to fewer auction rounds. Kwasnica et al. [19] propose an item-price auction called Resource Allocation Design (RAD), and find via lab experiments that it performs better than the baseline Simultaneous Multiple Round (SMR) auction [22]. The innovation of RAD over prior designs is to compute item prices that approximately clear the market. In separate laboratory experiments, Scheffel et al. [31] find that *i*Bundle, VCG, and linear price formats (including ALPS, a design inspired by RAD) all have similar allocative efficiency in environments with up to 18 items; they also observe that *i*Bundle requires many more rounds than linear price auctions.

The types of polynomial price auctions implemented in this work have been analyzed theoretically by Abernethy et al. [1]—without adaptive pricing—who provide convergence rate bounds that depend on the polynomial degree. With the exception of the *i*Bundle(d) variant that switches from anonymous to personalized prices [26], all auction formats in the literature have a fixed price structure. Day [11] studies a variety of other pricing schemes in CAs that can be obtained via the duality between allocation and pricing, and discusses their economic intepretation. It may be possible to embed some of these structures in our framework to yield other adaptive-price CAs.

2 BACKGROUND

We consider a model where a single seller holds a set of m distinct items which are auctioned among n agents (the buyers). The items are indivisible and there is unit supply of each item. Let M denote the set of items and N denote the set of agents. A *bundle* is a subset of items $Z \subseteq M$; we denote the set of bundles by $\mathcal{Z}$ $(= 2^M)$, which includes the empty bundle. The preferences of agent $i \in N$ are captured by its *valuation* function $v_i : \mathcal{Z} \to \mathbf{R}$. We assume that valuations are normalized and monotone: $v_i(\emptyset) = 0$ and $v_i(Z) \leq v_i(Z')$ when $Z \subseteq Z'$. We do not make any further assumptions on valuations as we develop our auction (although we will naturally impose some structure in our examples and empirical evaluation). An *allocation* is a vector of bundles $\mathbf{Y} = (Y_1, \ldots, Y_n)$ assigned to the agents. An allocation $\mathbf{Y}$ is *feasible* if $Y_i \cap Y_j = \emptyset$ for $i \neq j$; we denote the set of feasible allocations by $\mathcal{F}$. An allocation $\mathbf{Y}$ is *efficient* if

$$\mathbf{Y} \in \arg\max_{\mathbf{Y}' \in \mathcal{F}} \sum_{i \in N} v_i(Y_i'), \tag{1}$$

or in words, if it maximizes the total value to the agents among all feasible allocations. As we have formulated it so far, the efficient allocation problem is computationally intractable because the inputs themselves (the valuations) are of exponential size. Even when restricting the class of valuations so that each agent only values a single bundle (i.e., single-minded agents), computing the efficient allocation is NP-hard by reduction from weighted set packing [24]. Combinatorial auction implementations, both single-shot and iterative, therefore make use of integer programming techniques to solve these problems in practice [29]. Iterative auctions typically solve a revenue-maximization problem at each round, which is just the efficient allocation problem where valuations are replaced with prices. Depending on price structure, this may be simpler or just as computationally difficult as efficient allocation with respect to valuations. We return to this issue in Section 3.

2.1 Market Clearing

In running an iterative auction, the seller maintains *prices* for each agent $i \in N$. Like valuations, the prices are mappings $p_i : \mathcal{Z} \to \mathbf{R}$. As written, the prices may be *personalized*, meaning that different agents may face different prices; the vector of personalized prices is denoted $\mathbf{p} = (p_1, \ldots, p_n)$. Several bundle-price auctions proposed in the literature use personalized prices, including *i*Bundle, which will serve as a baseline for our experiments [26]. Prices are *anonymous* if $p_i = p_j$ for all $i, j \in N$; in this case we usually drop the agent index and simply write p to represent the prices.

We assume that agents have quasi-linear utility, so that agent i's utility for obtaining bundle Z at prices p_i is $u_i(Z; p_i) = v_i(Z) - p_i(Z)$. Agent i's *ϵ-demand set* at prices p_i is

$$D_i^\epsilon(p_i) = \{Z \in \mathcal{Z} : u_i(Z; p_i) \geq u_i(Z'; p_i) - \epsilon, Z' \in \mathcal{Z}\}\,. \tag{2}$$

In words, this is the set of bundles that maximize the agent's utility to within an additive error of ϵ. We write $D_i \equiv D_i^0$ to denote the agent's exact demand mapping. Given prices $\mathbf{p}$, the revenue of allocation $\mathbf{Y}$ is simply $\sum_{i \in N} p_i(Y_i)$. We assume that the seller does not value the items, so the utility it derives from an auction is the revenue collected. The seller's *supply set* at prices $\mathbf{p}$ is

$$S(\mathbf{p}) = \operatorname*{arg\,max}_{\mathbf{Y} \in \mathcal{F}} \sum_{i \in N} p_i(Y_i), \tag{3}$$

namely the set of feasible allocations that maximize the seller's revenue. We stress that (2) and (3) define set-valued mappings, not just selections of a single bundle or allocation.

An iterative auction proceeds over rounds, updating a provisional allocation along with prices at each round until there is a balance between demand and supply. To formalize this notion, we say that prices $\mathbf{p}$ are *market clearing* if

$$(\times_{i \in N} D_i(p_i)) \cap S(\mathbf{p}) \neq \emptyset. \tag{4}$$

This means that at prices $\mathbf{p}$, there exists a feasible allocation $\mathbf{Y} \in S(\mathbf{p})$ that maximizes the seller's revenue, such that Y_i maximizes agent i's utility for all $i \in N$. In this sense, $\mathbf{Y}$ represents a balance between demand and supply at prices $\mathbf{p}$. We define ϵ-market clearing prices analogously by replacing D_i with D_i^ϵ in (4). Given market clearing prices $\mathbf{p}$, we say that they *support* any allocation $\mathbf{Y}$ in the nonempty intersection in (4), and we call the pair $(\mathbf{Y}, \mathbf{p})$ an *ϵ-competitive equilibrium.*[1] The following standard result, which is used throughout the literature on combinatorial auctions [13, 23, 26], gives the key property of competitive equilibrium.

Proposition 2.1. [8] *If* $(\mathbf{Y}, \mathbf{p})$ *is an ϵ-competitive equilibrium, then* $\mathbf{Y}$ *is efficient to within an additive error of $n\epsilon$.*

This result motivates a natural termination criterion: the auction halts if the provisional allocation and current prices form an ϵ-competitive equilibrium, where ϵ is a discount set by the auctioneer. The criterion can be checked by offering an ϵ discount on the price of any bundle in the provisional allocation. If the buyers follow myopic best-response bidding and accept their allocated bundles, this confirms the demand-set condition of competitive equilibrium. Proposition 2.1 establishes that this termination rule comes with efficiency guarantees, which improve with smaller discount parameter ϵ. The discount parameter is used in practice (and in our evaluation) to control a trade-off between speed of convergence and efficiency of the final allocation.

2.2 Price Structure

Since the domain of bundles is exponential in size, an iterative auction cannot maintain explicit bundle prices even for with moderate number of items. We must therefore consider sparse encodings for prices. The classic approach is to specify item prices (also called linear prices), and the price of a bundle becomes the sum of its item prices.

In this work we will implement our adaptive-price auction using *polynomial* prices, a generalization of item pricing. These prices are formally specified as a pair $(\mathcal{M}, w)$ where $\mathcal{M} \subseteq \mathcal{Z}$ is a

[1]When the price function takes the form of item prices, this is called a Walrasian equilibrium. In this case, the condition $\mathbf{Y} \in S(\mathbf{p})$ is equivalent to ensuring items with positive price must all be allocated, and is usually stated this way [16].

collection of bundles and $w : \mathcal{M} \to \mathbf{R}$ is a function mapping the bundles to *coefficients*. The price of a bundle Z is then

$$p(Z) = \sum_{Z' \in \mathcal{M} : Z' \subseteq Z} w(Z'). \tag{5}$$

We require $w(\emptyset) = 0$ for normalization purposes. The set $\mathcal{M}$ captures the 'monomials' in the polynomial prices. In practice $\mathcal{M}$ should remain sparse. If we impose the constraint that $\mathcal{M}$ only consist of singleton bundles, then we recover linear prices. To understand why the structure in (5) corresponds to a polynomial, suppose that $\mathcal{M}$ only contains bundles of size at most two, which leads to *quadratic* prices. In this case, polynomial prices over three items $\{a, b, c\}$ can alternatively be expressed in the form

$$w(\{a\})x_a + w(\{b\})x_b + w(\{c\})x_c + w(\{a, b\})x_a x_b + w(\{a, c\})x_a x_c + w(\{b, c\})x_b x_c,$$

where $x_a, x_b, x_c \in \{0, 1\}$ are indicator variables for the items contained in the input set. This is a quadratic polynomial, with the coefficients given by w. The *degree* of the polynomial prices is $\max_{Z \in \mathcal{M}} |Z|$, and their *sparsity* is $|\mathcal{M}|$.

Besides item prices, the other most common structure is *bundle* prices. These are represented by a collection of bundle-value pairs captured in an expression of the form $(X_1, \alpha_1) \oplus \cdots \oplus (X_k, \alpha_k)$, which represents the following price function over bundles $Z \in \mathcal{Z}$:

$$p(Z) = \max \{\alpha_\ell : X_\ell \subseteq Z,\ \ell = 1, \ldots, k\}.$$

The $\oplus$ operator refers to the fact that only one of the X_ℓ bundles can be chosen.[2] For personalized prices, a separate expression is listed for each agent. The sparsity of bundle prices is k, the number of bundles in the price expression. In bundle-price auctions like *i*Bundle, an (X_ℓ, α_ℓ) term is added each time a new bundle X_ℓ is bid on by an agent.

2.3 Existence of Clearing Prices

The existence of market clearing prices with specific structure (e.g., linear, anonymous) depends on the class of agent valuations at hand. With general valuations, fully-expressive personalized prices are needed to ensure market clearing in the worst case [8]; fully-expressive prices can be encoded using bundle prices or m-degree polynomials, although in either case the prices may be exponential in size. If the valuations are super-additive, then anonymous prices suffice [27].[3] At the other extreme, anonymous linear prices are known to clear the market when agents have gross substitutes valuations [16, 18], which includes additive and unit-demand valuations as special cases. To simplify the exposition, all prices in the remainder of the paper are understood to be anonymous unless stated otherwise.

The following examples illustrate how it can be very difficult a priori to know what structure of prices is needed to clear the underlying valuations. With just a slight change to valuations, one can go from an instance with linear clearing prices, to an instance where prices must essentially be fully expressive to clear the market. In the examples, a bundle-value pair (X, α) represents a single-minded agent with valuation

$$v_i(Z) = \begin{cases} \alpha & \text{if } X \subseteq Z \\ 0 & \text{otherwise} \end{cases}$$

A single-minded valuation expresses complementarity among the items in X.

Example 2.2. There are three items $\{a, b, c\}$ and four single-minded agents with valuations $(\{a, b\}, 3)$; $(\{b, c\}, 3)$; $(\{a, c\}, 3)$; $(\{a, b, c\}, 5)$. The efficient allocation is to give all items to the last agent. It is easy to verify that this allocation is supported by linear prices of 1.5 for each item. If we change the last agent slightly to $(\{a, b, c\}, 4)$, then the efficient allocation remains unchanged,

[2] This corresponds to the XOR bidding language [24], but it can equally well be used to represent prices.

[3] A valuation v_i is super-additive if $v_i(Z \cup Z') \geq v_i(Z) + v_i(Z')$ whenever $Z \cap Z' = \emptyset$.

but it is an easy exercise to show that now linear and even quadratic clearing prices no longer exist. Third-degree (i.e., cubic) polynomial prices do clear the example. Since the valuations are super-additive, there exist anonymous bundle clearing prices [26], and with just three items a third-degree polynomial can reproduce these prices.

Example 2.3. A general counterexample that yields the inexistence of *anonymous* clearing prices is given by [25]. Their construction also yields examples where personalized clearing prices fail to exist whenever the prices are not fully expressive. Given valuation v_1, let the second agent's valuation be $v_2(Z) = v_1(M) - v_1(M \setminus Z)$. Nisan and Segal [25] show that the unique clearing prices are $p_1 = v_1$ and $p_2 = v_2$ with such valuations. The prices must therefore be personalized, and if there are m items, v_1 can be constructed so that it can only be represented by an m-degree polynomial.

3 ITERATIVE AUCTION

In this section we introduce our adaptive-price iterative auction design. Our auction is an instance of the subgradient class of iterative auctions [13]. We first introduce the main auction steps performed at each round, and then describe in detail the processes of computing a provisional allocation and performing price expansion.

3.1 Auction Steps

We first describe a non-adaptive version of the polynomial price auction where the polynomial terms are fixed. The auction proceeds over rounds indexed by t, which consist of *allocation*, *bidding*, and *price update* steps. There is also a *termination* criterion checked at each round. The auction maintains polynomial prices $(\mathcal{M}, w^t)$ where the set of monomials $\mathcal{M}$ is fixed for now, but the coefficients w^t are updated across rounds. The polynomial coefficients are initialized to 0, leading to zero initial prices. In this work we focus on the question of price structure, rather than personalized versus anonymous pricing, so we restrict our attention to anonymous prices for simplicity. We will briefly discuss later how our price expansion methodology can be adapted to introduce price personalization. As the auction prices are anonymous, we write p^t rather than $\mathbf{p}^t$ to refer to the prices. The auction is parametrized by a discount $\epsilon > 0$ and a step size schedule $\{\eta^t\}_{t=1}^{\infty}$ which controls the magnitude of the price updates as the rounds progress.

Allocation. At the beginning of each round t, the seller computes a provisional allocation $\mathbf{Y}^t$ that maximizes its revenue at current prices p^t, so that $\mathbf{Y}^t \in S(p^t)$. In the first round the prices are zero and the provisional allocation is empty by convention. Under polynomial prices, computing the provisional allocation can be straightforwardly formulated as an integer linear program using binary variables to encode which items each agent receives and to relate the items to the polynomial terms. However, the program consists of $\Theta(mns)$ constraints and $\Theta(nm + ns)$ variables, where s is the sparsity of the polynomial prices. Unless one uses just linear prices, this program scales too poorly with the sparsity of the polynomial to be of practical use. In Section 3.2 below we will describe a scalable approach to computing the provisional allocation which performs well in practice, at the expense of weaker theoretical efficiency guarantees.

Bidding. Next comes the bidding step. Given the current prices p^t and provisional allocation $\mathbf{Y}^t$, each agent i may choose to accept the bundle Y_i^t or bid by communicating a different bundle X_i^t. In the former case, the agent is essentially bidding on its allocated bundle, so we notate this as $X_i^t = Y_i^t$. When reasoning about its bid, we stipulate that the agent may take a discount of ϵ on the price of Y_i^t . Note that, following the usual clock auction format, an agent only communicates which bundle it wishes to bid on without a bid price.

Termination. If every agent $i \in N$ accepts the bundle Y_i^t it is allocated in the current round, or in other words if $X_i^t = Y_i^t$ for all $i \in N$, then the auction terminates.

Price Update. If the termination condition does not hold, then prices are updated for the next round. Following [1], the update rule for the polynomial prices takes the form

$$w^{t+1}(Z) = w^t(Z) + \eta^t |\{i \in N : Z \subseteq X_i^t\}| - \eta^t |\{i \in N : Z \subseteq Y_i^t\}|, \quad (6)$$

for all nonempty $Z \in \mathcal{M}$. The rule can be interpreted as an update to coefficient $w(Z)$ in the direction of *excess demand* for the combination of items Z: we count the number of agent bids that contain Z, and subtract away the number of allocated bundles that contain Z (there can be at most one of the latter). We then update the coefficient on Z by this quantity, weighted by the step size η^t. Note that when $X_i^t = Y_i^t$ for all $i \in N$ (i.e., the termination condition holds), the price update is zero. It is also possible for the price coefficient to be updated not only upwards but also downwards, and this translates into non-monotonic prices.

This iterative auction is designed to achieve high levels of efficiency when agents follow a bidding strategy known as *straightforward bidding*. Under this strategy, agent i's bid at round t satisfies $X_i^t \in D_i(p^t)$. However, if $Y_i^t \in D_i^\epsilon(p^t)$, then the agent chooses to accept its allocation because of the price discount. Recall that the provisional allocation is chosen such that $\mathbf{Y}^t \in S(p^t)$. Under straightforward bidding, the termination condition therefore amounts to verifying that $(\mathbf{Y}^t, p^t)$ constitutes an ϵ-competitive equilibrium, and therefore that $\mathbf{Y}^t$ is approximately efficient by Proposition 2.1.

Under straightforward bidding, the auction formally corresponds to a subgradient algorithm to solve the dual linear program to the efficient allocation problem—we refer to Abernethy et al. [1] for complete details. Subgradient algorithms for linear programs, and convex programs more generally, are only guaranteed to converge to an exactly optimal solution in the limit, rather than a finite number of steps. This is the reason for introducing the ϵ discount in practice. Other iterative auctions of the subgradient class, such as *i*Bundle, also use this device. A larger ϵ discount induces earlier convergence, at the expense of a weaker efficiency guarantee upon termination.

3.2 Provisional Allocation

We noted in the previous section that the problem of finding a revenue-maximizing allocation can be formulated as an integer program (IP), given polynomial prices. However, the size of the IP scales poorly with the price degree: with just 10 items and 5 agents the program requires hundreds of constraints under just quadratic prices. From a design standpoint, it is also undesirable to have a core part of the auction—provisional allocation—depend so closely on price structure.

To address both of these issues, we propose a generic approach to revenue maximization based on the idea of restricting the set of possible allocations. Because we no longer ensure $\mathbf{Y}^t \in S(p^t)$, this means we that lose the theoretical efficiency guarantee provided by Proposition 2.1. It is nonetheless still possible to provide a worst-case approximation guarantee, given below, and our experiments will show that the approach is very effective in practice.

Let $\mathcal{B}_i$ be the collection of bundles that have been bid on by agent i up to round t—the collection of 'observed' bundles, which includes the empty bundle by convention. We apply a natural restriction: the seller may only allocate to agent i a bundle from this collection, namely a bundle that has been explicitly bid on. More precisely, the seller is restricted to the set $\mathcal{F}' = \{\mathbf{Y} \in \mathcal{F} : Y_i \in \mathcal{B}_i,\ i \in N\}$ when computing the provisional allocation. Finding the revenue-maximizing allocation is still an integer programming problem, formulated as follows. For each $i \in N$ and $X_i \in \mathcal{B}_i$, we introduce indicator variable $x_i(X_i) \in \{0, 1\}$ to denote whether X_i is allocated to i in the current round.

$$\max_x \quad \sum_{i \in N} \sum_{X_i \in \mathcal{B}_i} p^t(X_i)\, x_i(X_i) \tag{7}$$

$$\text{s.t.} \quad \sum_{i \in N} \sum_{X_i \in \mathcal{B}_i,\, a \ni X_i} x_i(X_i) = 1 \qquad (a \in M) \tag{8}$$

$$\sum_{X_i \in \mathcal{B}_i} x_i(X_i) = 1 \qquad (i \in N) \tag{9}$$

As long as the number of rounds stays limited (i.e., fast enough convergence), this is a tractable problem for modern IP solvers. The number of constraints is constant at $m + n$, and the number of variables is nt. Importantly, the IP and its objective (7) in particular do not depend in any way on the price structure: one only needs to evaluate prices over the observed bundle for each agent.

The program itself represents a set packing problem, for which IP methods are particularly effective [2, 29], so we refer to this provisional allocation scheme as *set-packing* allocation. There is now the risk of terminating with an inefficient allocation because condition $\mathrm{Y}^t \in S(p^t)$ in (4) is not confirmed. On the other hand, restricting possible allocations can encourage early termination in practice. Although the true test of set-packing allocation lies in its empirical performance, some mild theoretical guarantees are possible. The following auxiliary result holds independently of price structure, for any iterative auction that follows the template described so far.

Proposition 3.1. *Under single-minded agents, an iterative auction with set-packing allocation obtains an efficient allocation upon termination. Under general valuations, it obtains an n-approximation to the efficient allocation upon termination.*

The result above presumes $\epsilon = 0$. For $\epsilon > 0$, the result holds with the usual adjustment of an additive error of $n\epsilon$ to the efficiency guarantee. The n-approximation is not a strong guarantee. Its main purpose is to confirm that under set-packing allocation, the efficiency of the auction upon termination cannot be arbitrarily bad. It is also important to stress that the result does not guarantee that the auction will terminate (the 'upon termination' state may not be reached)—this depends on the structure of prices and valuations.

3.3 Adaptive Pricing

The polynomial-price auction described so far keeps the monomial structure $\mathcal{M}$ fixed across all rounds. In practice, it can be difficult to know a priori whether linear prices will suffice to clear the market; on the other hand, using a high-degree polynomial is impractical for even a moderate number of items. Here we introduce an adaptive version of the polynomial price auction which augments the price structure as the auction progresses.

Cutting Planes. The key insight is that expanding the polynomial prices with a new monomial term corresponds to generating a cut for the associated allocation problem, towards computing an integer solution. We begin by formulating the efficient allocation problem as a linear program (as opposed to an integer program). We introduce a variable $x_i(X_i)$ as before for each $i \in N$ and $X_i \in \mathcal{B}_i$, but it is now *fractional*. We also introduce a fractional variable $y(\mathrm{Y})$ for each $\mathrm{Y} \in \mathcal{F}'$. Recall that $\mathcal{M}$ represents the current set of monomials in the polynomial prices. An LP formulation of the efficient allocation problem is as follows. With a slight abuse of notation, we write $\mathrm{Y} \supseteq Z$

when there is some Y_i in allocation $\mathbf{Y} = (Y_1, \ldots, Y_n)$ such that $Y_i \supseteq Z$.

$$\max_{x_i \geq 0,\, y \geq 0} \quad \sum_{i \in N} \sum_{X_i \in \mathcal{B}_i} v_i(X_i)\, x_i(X_i)$$

$$\text{s.t.} \quad \sum_{i \in N} \sum_{X_i \in \mathcal{B}_i, X_i \supseteq Z} x_i(S_i) = \sum_{\mathbf{Y} \in \mathcal{F}', \mathbf{Y} \supseteq Z} y(S) \qquad (Z \in \mathcal{M}) \tag{10}$$

$$\sum_{X_i \in \mathcal{B}_i} x_i(S_i) = 1 \quad (i \in N), \qquad \sum_{\mathbf{Y} \in \mathcal{F}'} y(S) = 1 \tag{11}$$

This LP is closely related to formulations given by Bikhchandani and Ostroy [8] and Abernethy et al. [1]. The dual to the allocation problem is a pricing problem, captured by the following LP.

$$\min_{\pi, w} \quad \sum_{i \in N} \pi_i + \pi^s \tag{12}$$

$$\text{s.t.} \quad \pi_i \geq v_i(X_i) - p(X_i) \quad (i \in N,\, X_i \in \mathcal{B}_i) \tag{13}$$

$$\pi^s \geq \sum_{i \in N} p(Y_i) \qquad (\mathbf{Y} \in \mathcal{F}') \tag{14}$$

where $p(X)$ above is short-hand for $\displaystyle p(X) = \sum_{Z \in \mathcal{M} : Z \subseteq X} w(Z),$

which is the desired polynomial structure. We refer to the primal allocation LP as (P) and its dual pricing LP as (D). Using standard duality arguments from Bikhchandani and Ostroy [8], one can show that polynomial prices with the current structure $\mathcal{M}$ can clear the market if and only if (P) has an *integer* optimal solution. The clearing prices can be obtained from the optimal solution to (D): the dual variables $w(Z)$ associated with each constraint in (10) provide the monomial coefficients. From this perspective, expanding the prices by introducing another monomial Z is equivalent to introducing a constraint in (10), strengthening the formulation and cutting off fractional solutions.

The linear program (P) therefore provides, at least in principle, a test to check whether the current price structure is expressive enough to support an efficient allocation: solve it and check whether the solution is integer. The problem is, of course, that the valuations v_i in the objective are not available to the auctioneer in the first place.

Restricted Primal. To test the clearing ability of the current price structure at any given round, we instead work with an alternative LP called the *restricted primal* (RP). This LP shares the same variables and feasible set (10–11) as (P), but its objective is replaced with

$$\max_{x_i \geq 0,\, y \geq 0} \quad \sum_{i \in N} \sum_{X_i \in D_i(p^t)} x_i(X_i) + \sum_{\mathbf{Y} \in S'(p^t)} y(\mathbf{Y}) \tag{15}$$

where the mapping S' is defined analogously to the supply set in (3), with the restriction that $\mathbf{Y} \in \mathcal{F}'$. Given the agents' demand sets at the current prices p^t, this objective can in principle be formulated. There is still the difficulty that the supply set $S'(p^t)$ could be large and difficult to enumerate. Instead, in practice we can use the standard technique of column generation to solve the LP without enumerating all the $y(\mathbf{Y})$ variables [6, Chap. 6]. This involves finding a variable $y(\mathbf{Y})$ with positive reduced cost, which can be found via an integer program very similar to the one used to compute the provisional allocation. We defer the specific implementation details to the appendix.

The crucial property of (RP) is that it allows us to test whether the original formulation (P) has an integer optimal solution, assuming that the current prices p^t have converged to an optimal solution for (D).

Theorem 3.2. *Let (π, w) be a feasible solution to (D), used to obtain demand and supply sets and form (RP). Then (x, y) and (π, w) are optimal solutions to (P) and (D), respectively, if and only if (x, y) is an optimal solution to (RP) with value $n + 1$.*

Proof. Note that given variables w defining p in the dual (D), the setting of the other variables is implied since at the optimum we must have $\pi_i = \max_{X_i \in \mathcal{B}_i} v_i(X_i) - p(X_i)$ for all $i \in N$, and $\pi^s = \max_{Y \in \mathcal{F}'} \sum_{i \in N} p(Y_i)$. Therefore, in reasoning about (D), it is sufficient to know w only, and the remaining variables are implied.

First recall the complementary slackness conditions. Let (x, y) and (π, w) be feasible solutions to (P) and (D) respectively. Then they have the same objective value if and only if

$$x_i(X_i) > 0 \Rightarrow \pi_i = v_i(X_i) - p_i(S_i) \Rightarrow X_i \in D_i(p)$$
$$y(Y) > 0 \Rightarrow \pi^s = \sum_i p_i(Y_i) \Rightarrow Y \in S'(p)$$

where p is the price function defined by coefficients w. By weak duality, the value of (P) is always no more than the value of (D), so this implies that both solutions are optimal.

Note that (P) and (RP) share the same feasible set, so a feasible solution for one is feasible for the other. To prove the forward direction, let (x, y) be an optimal solution to (RP) of value $n + 1$, formed from feasible (D) solution (π, w). We have that

$$\sum_{X_i \in D_i(p)} x_i(X_i) \leq \sum_{X_i \in \mathcal{B}_i} x_i(X_i) = 1$$

for each agent i and

$$\sum_{Y \in S'(p)} y(Y) \leq \sum_{Y \in \mathcal{F}'} y(Y) = 1$$

As the objective value of (RP) is $n + 1$, each of the inequalities above must hold with equality. Thus for any i and $x_i(X_i) > 0$, the variable must contribute to the objective, and $X_i \in D_i(p)$. An analogous argument for the seller shows that $y(Y) > 0$ implies $Y \in S'(p)$. Therefore (x, y) and (π, w) satisfy complementary slackness, and are optimal for (P) and (D) respectively.

For the reverse direction, assume (x, y) and (π, w) are optimal for (P) and (D) respectively. As mentioned, (x, y) is also feasible for (RP). By complementary slackness, $x_i(X_i) > 0 \Rightarrow X_i \in D_i(p)$, and therefore

$$\sum_{X_i \in D_i(p)} x_i(X_i) = \sum_{X_i \in \mathcal{B}_i} x_i(X_i) = 1$$

Similarly, $y(Y) > 0 \Rightarrow Y \in S'(p)$ and therefore

$$\sum_{Y \in S'(p)} y(Y) = \sum_{Y \in \mathcal{F}'} y(Y) = 1$$

This implies that the objective value of (RP) at solution (x, y) is $n + 1$, which must be optimal. □

Note that the theorem makes no assumptions on whether (x, y) is integral. The following corollary leads to a test for price expansion.

Corollary 3.3. *Form (RP) based on the current prices p^t. If the solution to (RP) has optimal value strictly less than $n + 1$, then prices p^t are not market clearing. If the solution to (RP) has optimal value $n + 1$ attained at an integer solution, then prices p^t are market clearing.*

Proof. Let w^t be the current price coefficients, and complete the feasible dual solution to (π^t, w^t). Suppose that the optimal solution (x, y) to (RP) has value strictly less than $n + 1$, and assume for the sake of contradiction that p^t are market clearing. By complementary slackness, this implies that

(π^t, w^t) is an optimal solution to (D). Let (x', y') be an optimal solution to (P). Then by Theorem 3.2, (x', y') is an optimal solution to (RP) with value $n+1$. But this contradicts the premise that (x, y) was an optimal RP solution with value strictly less than $n+1$. Thus (π^t, w^t) cannot be optimal for (D), and p^t are not market clearing.

For the second statement, suppose (x, y) is an integer optimal solution to (RP) with value $n+1$. Then by Theorem 3.2, (x, y) and (π^t, w^t) are optimal solutions to (P) and (D) respectively. By complementary slackness, the prices p^t are therefore market clearing. □

To be explicit, the corollary dictates that we use the solution to (RP) to test for the need for price expansion. There are three cases.

(1) If the optimal solution to (RP) has value strictly less than $n+1$, then the auction proceeds by updating prices as in any other round—the fact that p^t are not market clearing ensures that the price update is nonzero.
(2) If the optimal value is $n+1$ and the solution to (RP) is integer, then the auction has converged to clearing prices.
(3) There remaining case occurs when the optimal value is $n+1$, but the solution to (RP) is fractional. In this case, we can make progress by cutting off the fractional solution by adding extra constraints in (10), which is equivalent to price expansion, as discussed.

The preceding results on integrality detection were proved in the context of polynomial prices for concreteness, but in fact the proofs are not tied to this structure and would immediately extend to other price structures.[4] The key property is that the price structure should arise, via duality, from a set of feasibility constraints replacing (10) relating demand-side variables x_i to supply-side variables y. Day [11] discusses many interesting alternatives along these lines. Expanding prices (i.e., generating a cut) in case (3) above does of course depend on a specific choice of price structure, and we address this next for polynomial prices.

Price Expansion. We propose a straightforward method for generating a cut. Let (x, y) be the optimal solution to (RP), and assume it is fractional, meaning the value of at least one variable lies in $(0, 1)$. Let Z be a monomial whose associated constraint in (10) contains a fractional variable. For a feasible allocation $\mathrm{Y} \supseteq Z$, let $\mathrm{Y}|_Z$ denote the unique bundle Y_i in Y such that $Y_i \supseteq Z$. (Such a bundle is unique as bundles in a feasible Y are pairwise disjoint.) We consider the set of candidate monomials

$$C = \{X_i : x_i(X_i) > 0,\ X_i \supseteq Z,\ X_i \in \mathcal{B}_i\} \cup \{\mathrm{Y}|_Z : y(\mathrm{Y}) > 0,\ \mathrm{Y} \supseteq Z,\ \mathrm{Y} \in \mathcal{F}'\}$$

and form the corresponding constraint for each in turn. If during this process we find a constraint that solution (x, y) violates, we perform price expansion by adding the monomial to $\mathcal{M}$. If the process fails to find a cut after iterating over all the constraints of (RP), then we can establish the following fact.

Theorem 3.4. *Assume that the optimal solution (x, y) to (RP) is fractional and has value $n+1$. If price expansion fails to generate a cut, then anonymous clearing prices do not exist.*

Proof. Suppose that the optimal solution to (RP) is fractional and that the price expansion process fails to find a cut for all constraints associated with current monomials $Z \in \mathcal{M}$. Then observe that we can introduce *all* possible monomials $Z \subseteq M$ into $\mathcal{M}$, and solution (x, y) still remains feasible for (RP). By Theorem 3.2, (x, y) is thus an optimal solution to (P) where constraints for all $Z \subseteq M$ are present. On the other hand, as the provisional allocation does not balance supply and demand, an integer solution to (P) does not exist. The LP formulation (P) with constraint for

[4]It is possible to prove the results at the level of an abstract "price structure", but this does not provide any additional insights and comes at the cost of significant extra notation.

all $Z \subseteq M$ is equivalent to the second-order formulation from Bikhchandani and Ostroy [8]. By their Theorem 2, anonymous clearing prices exist if and only if their second-order formulation has an integer optimal solution, which completes the result. □

If price expansion fails to find a cut, then personalized price are needed for continued progress. The auction can switch to personalized prices by instantiating n copies of the current price function. (More sophisticated approaches are possible where prices become personalized for just a subset of the agents.) The combination of price expansion and personalization is guaranteed to find a cut, since personalized clearing prices are guaranteed to exist. For the purposes of this paper, we restrict ourselves to price expansion in our implementation, and halt the auction with a flag of "personalization required" whenever the auction fails to generate a cut.

There may be several monomials $Z \in C$ that would generate a cut. We could introduce all of them, or to be more conservative use a heuristic to select one among them. For our evaluation we considered several heuristics: choosing a Z with minimum $|Z|$; choosing a Z with maximum $|Z|$; and choosing a Z with maximum absolute constraint violation. These all performed very similarly so we will report results using the latter heuristic.

4 EXPERIMENTAL SETUP

4.1 Domain Generators

The envisioned advantage of an adaptive-price auction is that it should work well across a range of application domains and a variety of valuation structures on the part of buyers, which may not be known a priori. Along these lines, we make use of two different generators to obtain valuation profiles for our experiments. The first is the Combinatorial Auction Test Suite (CATS), a standard generator that has been used extensively to test both single-shot and iterative auction designs [21]. The second is the Quadratic Model (QM) originally proposed by de Vries and Vohra [14], which was motivated by an analysis of bidding in spectrum auctions [4].

For all our auction implementations (including the benchmarks), interaction between the seller and buyers occurs solely through a *demand query* interface [9]. In a demand query, the seller quotes a buyer a price function (using some encoding like bundle or polynomial prices), and the buyer responds with a utility-maximizing bundle at the quoted prices in accordance with straightforward bidding.[5] Depending on the structure of valuations and prices, this may involve complicated optimization logic on the part of the buyer, but from the perspective of the auction this is a black box.

Multi-Minded Valuations. CATS generates instances from the class of multi-minded valuations, which take the form $(X_1, \alpha_1) \oplus \cdots \oplus (X_k, \alpha_k)$. CATS allows one to specify the number of items, but not the number of bids k per valuation. Instead, one specifies the total number of bids across all valuations in the generated profile. With multi-minded valuations, demand queries are conveniently computed, given any price representation, by evaluating the prices of the k bundles and finding the most preferred bundle in the collection. We assume that a multi-minded agent only ever bids on one of its k bundles of interest (i.e., it never includes extraneous items in its bid).

CATS instances are meant to be economically motivated, representing stylized models of allocation problems involving truck routes, network bandwidth, pollution rights, or real estate. We use the default CATS parameter settings to generate the instances throughout. We consider three generator distributions for our experiments: `regions`, `arbitrary`, and `paths`. We did not consider

[5]For tie-breaking purposes, the demand query might also include a proposal bundle X, which the agent must return as the response to the query if X maximizes its utility. We use this variant in our implementation, along with the stipulation that the agent may take an ϵ-discount on the price of X when computing the best-response.

the `scheduling` distribution in CATS because it is known to generate instances that are very easy to clear with linear prices [21].

Quadratic Valuations. In their survey of methods for single-shot CAs, de Vries and Vohra [14] proposed the *Quadratic* model of agent valuations. These valuations are encoded by specifying a value w_a for each item $a \in M$ as well as a 'synergy set' $\Gamma \subseteq M$. The value of a generic bundle $Z \in \mathcal{Z}$ is evaluated as

$$v_i(Z) = \sum_{a \in Z} w_a + \mu_i \sum_{a,b \in \Gamma \cap Z} w_a w_b. \qquad (16)$$

Just like quadratic prices, Quadratic valuations can be represented by a degree 2 polynomial. The Quadratic model is based on Ausubel et al. [4]'s analysis of FCC spectrum auction data, which found that complementarity between pairs of licenses is captured by the product of their populations.

In our experiments we use $\mu_i = 1/2$ and also hold the size the synergy set constant at $m/2$ throughout. As others have noted, one concern with Quadratic valuations is that an excess of synergy can lead to uninteresting efficient allocations with just one winner [7]. To mitigate this, one can work with *capped* Quadratic valuations of the form $v_i(Z;\kappa) = \max\{v_i(Z') : Z' \subseteq Z,\ |Z'| \leq \kappa\}$, where $1 \leq \kappa \leq m$ is a parameter bounding the maximum size of a bundle bid on. In our experiments we used a value of $\kappa = m/2$ for this cap. Computing the value of a bundle given a Quadratic or capped Quadratic valuation is fast and straightforward by evaluating (16). However, computing the response to a demand query given prices requires mixed-integer programming.

To summarize, in our experiments we use $\mu_i = 1/2$, a cap of $\kappa = m/2$, and synergy set size $|\Gamma| = m/2$ throughout. The synergy set Γ is drawn uniformly from M, while each item coefficient w_a for $a \in M$ is is drawn uniformly from $[0, 1]$, following de Vries and Vohra [14].

4.2 Baseline Mechanisms

In our experiments section we evaluate our adaptive-price auction emperically both in absolute terms, and by comparison to a suite of baseline mechanisms.

For our first comparison we use *i*Bundle[26].[6] We select *i*Bundle, not only because it is widely cited in the literature, but also because it is an exemplar of a mechanism with pure bundle pricing. Due to the expressiveness of bundle prices, *i*Bundle is known to perform well in settings with multi-minded combinatorial bidders with strongly peaked preferences, precisely the type of valuations produced by CATS. However, as we will see, *i*Bundle can struggle in settings where valuations are less peaked, such as the Quadratic generator. We consider the *i*Bundle variant with personalized bundle prices, which give *i*Bundle a further advantage over the other mechanisms that all use anonymous prices. The use of personalized bundle prices provides a guarantee that approximate clearing price can be computed by *i*Bundle in a finite (but possibly very large) number of rounds. Because we use anonymous prices for all the other auctions designs, it is possible that they fail to converge (which is not uncommon).

For our second baseline we implemented the clock phase of the CCA [3], a design which has been widely used in practice (e.g., the 2014 Canadian 700 MHz Auction, which raised more than $5 Billion CAD [17]). Because we are interested in studying the properties of iterative pricing mechanisms, we omit the CCA's last-and-final package phase. Such a phase could be bolted onto any of the mechanisms proposed here, and would likely further improve the results. But this would conflate the effect of the final round with the capability of the iterative phase and complicate interpretation of the results. To emphasize that we are focused on the clock phase, we call this mechanism the Linear Clock auction to highlight its price structure. One challenge in implementing an ascending clock auction is to determine the appropriate clock step size. Set the size too small, the auction will

[6]For *i*Bundle the bid discount ϵ coincides by design with the (constant) price increment.

Generator	IBundle	Linear Clock	Linear Exact	Linear Packing	Adaptive
Arbitrary	**100** (Cleared)	0 (Under Demand)	3.3 (Max Rounds)	20.0 (Max Rounds)	**100** (Cleared)
Paths	**100** (Cleared)	0 (Max Rounds)	26.7 (Max Rounds)	80.0 (Cleared)	40 (Personal)
Regions	**100** (Cleared)	0 (Under Demand)	6.7 (Max Rounds)	50.0 (Max Rounds)	**100** (Cleared)
Quadratic	0 (Max Time)	0 (Under Demand)	90.0 (Cleared)	63.3 (Cleared)	**100** (Cleared)
Mean	75	0	31.7	53.3	**85**

Table 1. Percent of auctions cleared for ϵ =0.05 and 30 items. The best value in each row is bold. Entries represent the mean of 30 runs.

take too many rounds. Set it too large, the prices my overshoot, resulting in under-demand. For our baseline we implement the approach commonly used in practice that uses an *increasing* step size [3], as a heuristic for finding reasonable increments over the rounds.

Next, we turn to analyzing our own mechanism. Recall the two major components of our design: set-packing allocation, and adaptive pricing. We first seek to capture the effect of set-packing in isolation by studying it in simpler mechanisms that omit the adaptive pricing aspect. To this end, we include a linear-price mechanism that maximizes revenue over the entire set of feasible allocations, which is computationally feasible under linear prices (i.e., a basic auction without either set-packing allocation or adaptive pricing). We compare this to a linear-price mechanism that employs our set-packing allocation (but not adaptive pricing). We call these mechanisms *Linear Exact* and *Linear Packing*, respectively. We note that the guarantee of Proposition 2.1 applies to the linear Exact variant if it converges, but either variant may fail to converge if linear prices are not expressive enough to clear the market. Finally, we include our actual proposed *Adaptive* mechanism.

4.3 Parametrization and Computation

All of the auctions we implemented make use of a discount ϵ on provisional allocation prices; a smaller ϵ should achieve higher efficiency, according to Proposition 2.1, but possibly at the expense of slower convergence. For our adaptive-price auction (and the linear-price variants) we use a diminishing step size schedule $\eta^t = cV/\sqrt{t}$, where V is the median bid value, for multi-minded valuations generated by CATS, and the largest bundle value across agents, for Quadratic valuations. The c parameter is a scaling factor, which we evaluated at the settings of 0.25%, 0.5%, 1%, 2%, 4%, 8%, 16%. When presenting the results, we select the c scaling parameter for each auction that yields the highest efficiency. This varied among the auctions and was typically on the order of 0.25% for the Linear Clock and 2% for the other auctions (scaling does not apply to *i*Bundle, which sets its step size based on the discount ϵ). Furthermore, in our adaptive-price auction we test for the need for price expansion after a fixed number of rounds called an *epoch*. We evaluated epoch lengths of 1, 5, 10, and 20 and report on results using an epoch of 10. For all auctions we imposed a cap of 1000 on the maximum number of rounds.

We use 30 items for all our auction instances. At this scale it is impractical to encode a general valuation by listing the value of every possible bundle, because there are over a billion bundles. However, as previously mentioned, CATS provides multi-minded valuations of tractable size. We generated 150 bids in all our CATS instances. For Quadratic instances we generated 5 agents. We ran 30 auction instances for all valuation domains. The auctions were implemented in Python 2.7, and the IPs were solved using CPLEX 12 on a 12-core 2.93Ghz Xeon machine.

5 RESULTS

In this section we evaluate our adaptive-price auction empirically both in absolute terms, and by comparison to the suite of baseline mechanisms described in Section 4.2.

Generator	IBundle	Linear Clock	Linear Exact	Linear Packing	Adaptive
Arbitrary	**99.6** (0.2)	52.1 (2.8)†	49.1 (4.7)†	81.4 (2.7)†	96.9 (0.6)
Paths	**99.8** (0.0)	87.0 (0.7)†	88.4 (1.7)†	99.7 (0.1)	98.6 (0.4)†
Regions	**99.4** (0.2)	51.5 (2.4)†	37.3 (5.7)†	93.0 (1.9)†	98.9 (0.3)
Quadratic	72.9 (0.9)†	82.4 (1.5)†	**96.1** (1.5)	86.2 (3.6)	94.7 (1.4)
Mean	92.9	68.3	67.7	90.1	**97.3**

Table 2. Efficiency of the auctions for ϵ =0.05 and 30 items. The best value in each row is bold. Entries are the mean of 30 runs with standard errors in parentheses. A † indicates less than half the instances cleared.

Clearing. Table 1 shows the percentage of instances that reached clearing prices for each mechanism under each of our four demains of study. We track not only the clearing percentage, but also supply the modal reason for any clearing failure, detailed below.

We observe that *i*Bundle is able to clear all of the CATS distributions, which is expected since it was designed for these types of settings. However, it is unable to clear any of the Quadratic instances, where short multi-minded representations do not exist and the set of bundles upon which agents bid becomes very large. This is a known failure mode of bundle price auctions, which can arise even with additive valuations. Parkes [26] proposes a variant of *i*Bundle that accepts non mutually-exclusive bids (i.e., OR bids), which allows it to handle additive valuations; however, Quadratic valuations do not necessarily have short descriptions as OR bids either.

Unlike the other auctions, the Linear CCA does not necessarily balance supply and demand upon termination, which we record as 'Under Demand' when it occurs. This was anticipated by the CCA's designers (recall the Linear Clock mechanism is the first phase of the CCA mechanism), which is why the full CCA includes a supplementary round. Here, though, we are interested in the informativeness of the iterative phase prices; these results provide some evidence that they are not as informative as our alternatives. We note that the Linear Clock auction terminates after 1000 rounds (our cap) in Paths when using the step-size settings producing the highest efficiency.

The Linear Exact mechanism is able to clear only a modest number of the CATS instances, given its restrictive pricing structure, with the balance of instances running out of rounds. However, it performs much better on the Quadratic valuations where it clears 90% of the instances. By contrast, the Linear Packing auction clears a much larger percentage of the CATS instances. This shows, perhaps surprisingly so, that the allocative restriction we deploy in the set-packing, for computational reasons, can actually result in a *higher* clearing rate.

Finally, we turn to our Adaptive auction, which clears 100% of the instances in Arbitrary, Regions, and Quadratic. It is never easy to compare mechanisms across multiple domains; here we adopt the mean as a simple metric. This is conservative, in that it overweights the CATS domains which play to *i*Bundle's strength. Even with this weight handicap, Adaptive still significantly outperforms the other mechanisms' clearing rate.

One does observe, however, that Adaptive clears only 40% of the Paths instances. We emphasize that in our mechanism, unlike the others, we know precisely why this is: anonymous prices are not sufficient to clear the market per Theorem 3.4. To clear Paths, it turns out that one typically needs nonlinear but also personalized prices. In many settings the use of anonymous prices is a hard constraint, so we opt to terminate our auction with these anonymous non-linear prices. The question then becomes how efficient is the outcome upon termination, which we turn to next.

Efficiency. We consider the distribution of final efficiency levels in Table 2. As expected, *i*Bundle is highly efficient on the CATS instances, but struggles achieving only 72.9% efficiency on Quadratic; overall it achieves on average 92.9% efficiency. The monotone price update restriction in the Linear Clock auction causes it to fare poorly in all settings, achieving an average of only 68.3% efficiency.

Generator	IBundle	Linear Clock	Linear Exact	Linear Packing	Adaptive
Arbitrary	**55.3** (2.3)	694.3 (7.7)†	998.4 (1.6)†	824.8 (66.3)†	125.3 (20.8)
Paths	**59.7** (2.5)	1000.0 (0.0)†	830.2 (57.8)†	545.4 (49.1)	319.6 (25.6)†
Regions	**57.7** (2.4)	682.5 (9.3)†	947.2 (37.8)†	655.8 (73.6)†	218.4 (27.8)
Quadratic	834.9 (1.9)†	87.4 (1.4)†	121.3 (54.5)	393.6 (85.8)	**47.2** (8.4)
Mean	251.9	616.1	724.3	604.9	**177.6**

Table 3. Rounds of the auctions for ϵ =0.05 and 30 items. The best value in each row is bold. Entries are the mean of 30 runs with standard errors in parentheses. A † indicates less than half the instances cleared.

We note that the real-world CCA has a second phase to help ameliorate the effects of inefficiencies in the first phase, but our goal here is to achieve high efficiency in a simpler, single phase setup.

The Linear Exact auction shines in the Quadratic setting, where it achieves the highest efficiency of all the mechanisms, 96.1%. However, like Linear Clock, it has poor performance on the CATS domains, giving it an overall average of only 67.7%. By contrast, the Linear Packing auction performs significantly better in the CATS domains and worse on the Quadratic domain, yielding an overall efficiency of 90.1%. Our Adaptive auction does not obtain the highest efficiency in any single domain. But, importantly, its consistentcy across different types of domains makes it the best on average at 97.3%, which is a significant lift over *i*Bundle.

Rounds and Runtime. For an auction to be practical it must terminate in a reasonable number of rounds, or the bidding process takes too long for participants. In practice, there is a trade-off between having a modest number of rounds through aggressive price updates, and having high effiency. Experimentally, we have parametrized our auctions for efficiency. Table 3 reports on the number of rounds required by each auction under the parameters yielding highest efficiency. As expected, *i*Bundle requires only a very modest number of rounds in CATS, but an immense number for Quadratic. By contrast, the Linear Clock and Linear Exact auctions both require only a modest number of rounds in Quadratic, and a very large number in CATS. The Linear Packing auction requires a relatively large number across domains. Finally, the Adaptive auction requires the smallest number in Quadratic, and only a modest number in CATS. It thus uses the smallest number of rounds on average, at 177.6. It is worth pointing out that the number of rounds is also closely related to the runtime needs to compute the auction. Although there are several nested optimization problems that the auctioneer must solve in the Adaptive auction (due to constraint generation), it still achieves the smallest overall runtime because of its low number of rounds, on average needing only 131.2 seconds to clear a market. Linear Clock needed 1188.1 and *i*Bundle needed 2710.1 seconds on average.

Revenue. Table 4 shows the revenue generated under each of the auctions and settings, listed as a percentage of the optimal social welfare. Unlike the previous metrics, there is no ideal level for revenue as it simply represents a split of the surplus between the buyers and seller. This said, we observe that *i*Bundle charges on average lower prices with a 79.4% average revenue, and Adaptive prices somewhat more aggressively with an average revenue of 87.0%.[7]

Price Structure. Finally, to provide some detail on how the Adaptive auction achieves the clearing, efficiency and speed we have seen so far, we examine the structure of the non-linear prices it constructs. Figure 1 shows the distribution of the highest degree of the polynomial prices used, across each of the domains. We observe that the auction uses an average degree of 8.8 across

[7]Because some auctions terminate with under demand (either by design in the case of Linear Clock, or by running out of rounds in the case of Linear Exact and Packing), we observe values larger than 100%. The final prices in these cases are not individually rational, which is what enables the extraordinarily high revenue.

Generator	IBundle	Linear Clock	Linear Exact	Linear Packing	Adaptive
Arbitrary	89.2 (0.8)	124 (1.5)†	109.3 (1.3)†	105.1 (1.7)†	93 (0.7)
Paths	78.2 (0.9)	21.3 (0.4)†	82.2 (0.9)†	80.5 (0.9)	78.8 (1.1)†
Regions	82.7 (1.1)	111.1 (1.6)†	91.3 (1.0)†	89.3 (1.0)†	87.5 (0.9)
Quadratic	67.4 (1.1)†	98 (2.2)†	92.4 (1.1)	92.7 (1.1)	88.6 (1.0)
Mean	79.4	88.6	93.8	91.9	87

Table 4. Revenue of the auctions for ϵ =0.05 and 30 items. Entries are the mean of 30 runs with standard errors in parentheses. A † indicates less than half the instances cleared.

domains. However in the Paths domain it uses only much smaller degree prices. As we have observed, Paths requires (often unrealistic) personalized prices to clear, and thus for this domain a high degree of non-linearity is not the limiting factor.

In addition, we are also interested in knowing how many non-linear terms participants must consider when bidding in the auction, as illustrated in Figure 2. We observe that the auction requires on average 69.2 terms in the Paths domain (it adds more, relatively lower degree terms there) and on average 53.8 terms in the rest of the auctions. We note that 30 of these terms are the standard item prices (there are 30 goods in our domains). So the auction typically only adds a modest number of non-linear offsets to the linear prices in order to achieve the significant improvement in performance we observed previously. The plot also includes data on the number of bundle price terms required by *i*BundleḞor the CATS domains this is a reasonable but comparatively large number at 150 on average; we omit the Quadratic data in the figure, where *i*Bundle times out after adding 3339 terms on average.

6 CONCLUSIONS

This work introduced the first iterative CA design with fully adaptive pricing, along with a practical implementation of the auction using polynomial prices. The design provably detects the need for price expansion to ensure market clearing. Our empirical evaluation showed that the auction is simultaneously competitive with *i*Bundle under multi-minded valuations, and linear-price auctions under quadratic valuations, in terms of clearing, efficiency, and convergence rate, making it a method of choice when information about buyer valuation structure is not available a priori.

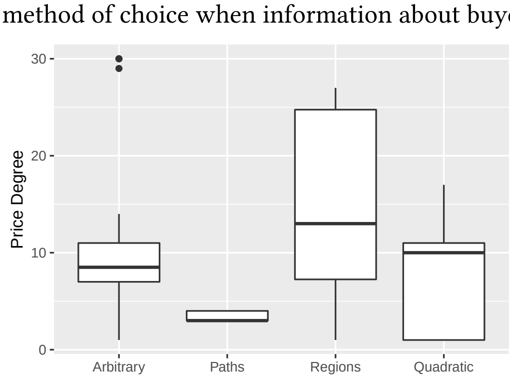

Fig. 1. Box plot showing the distribution of the degree of the prices in the Adaptive auction with ϵ =0.05. Each box represents 30 samples in a domain with 30 goods.

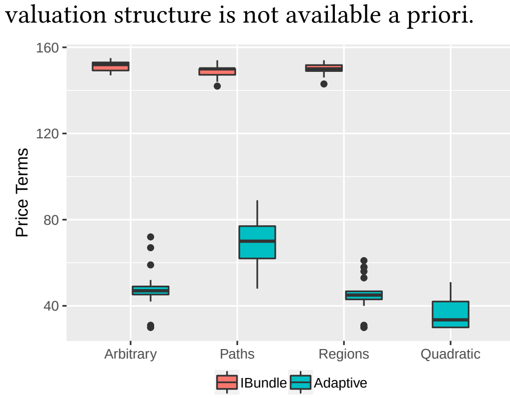

Fig. 2. Box plot showing the distribution of the number of terms in the prices of IBundle and the Adaptive auctions (the others are linearly priced) with ϵ =0.05. IBundle fails to clear instances in the Quadratic domain, and we therefore omit this data point. Each box represents 30 samples in a domain with 30 goods.

REFERENCES

[1] Jacob Abernethy, Sébastien Lahaie, and Matus Telgarsky. 2016. Rate of Price Discovery in Iterative Combinatorial Auctions. In *Proc. 17th Conf. on Econ. and Comp.* 809–809.

[2] Arne Andersson, Mattias Tenhunen, and Fredrik Ygge. 2000. Integer programming for combinatorial auction winner determination. In *MultiAgent Systems, 2000. Proceedings. Fourth International Conference on.* IEEE, 39–46.

[3] Lawrence M. Ausubel and Oleg V. Baranov. 2014. *A Practical Guide to the Combinatorial Clock Auction.* Technical Report. University of Maryland.

[4] Lawrence M Ausubel, Peter Cramton, R Preston McAfee, and John McMillan. 1997. Synergies in wireless telephony: Evidence from the broadband PCS auctions. *J. Econ. & Mgmt. Strat.* 6, 3 (1997), 497–527.

[5] Lawrence M. Ausubel and Paul R. Milgrom. 2002. Ascending auctions with package bidding. *Adv. Theo. Econ.* 1, 1 (2002).

[6] Dimitris Bertsimas and John N. Tsitsiklis. 1997. *Introduction to Linear Optimization.* Vol. 6. Athena Scientific.

[7] Martin Bichler, Pasha Shabalin, and Alexander Pikovsky. 2009. A computational analysis of linear price iterative combinatorial auction formats. *Info. Sys. Res.* 20, 1 (2009), 33–59.

[8] Sushil Bikhchandani and Joseph M. Ostroy. 2002. The package assignment model. *J. Econ. Theory* 107, 2 (2002), 377–406.

[9] Liad Blumrosen and Noam Nisan. 2009. On the computational power of demand queries. *SIAM J. Comput.* 39, 4 (2009).

[10] Peter Cramton. 2013. Spectrum auction design. *Review of Industrial Organization* 42, 2 (2013), 161–190.

[11] Robert Day. 2018, June. Linear Prices in Combinatorial Auctions. Paper presented at the INFORMS Workshop on Mathematical Optimization in Market Design, Ithaca, NY.

[12] Robert Day and Peter Cramton. 2012. Quadratic core-selecting payment rules for combinatorial auctions. *Op. Res.* 60, 3 (2012), 588–603.

[13] Sven de Vries, James Schummer, and Rakesh V. Vohra. 2007. On ascending Vickrey auctions for heterogeneous objects. *J. Econ. Theory* 132, 1 (2007), 95–118.

[14] Sven de Vries and Rakesh V. Vohra. 2003. Combinatorial auctions: A survey. *INFORMS J. on comp.* 15, 3 (2003), 284–309.

[15] A. Goetzendorf, M. Bichler, P. Shabalin, and R. W. Day. 2015. Compact Bid Languages and Core Pricing in Large Multi-item Auctions. *Management Science* 61(7) (2015), 1684–1703.

[16] Faruk Gul and Ennio Stacchetti. 1999. Walrasian equilibrium with gross substitutes. *J. Econ. Theory* 87, 1 (1999), 95–124.

[17] Industry Canada. 2015. Canadian 700MHz Auction. http://www.ic.gc.ca/eic/site/smt-gst.nsf/eng/h_sf10598.html.

[18] Alexander S. Kelso and Vincent P. Crawford. 1982. Job matching, coalition formation, and gross substitutes. *Econometrica* 50, 6 (1982), 1483–1504.

[19] Anthony M Kwasnica, John O Ledyard, Dave Porter, and Christine DeMartini. 2005. A new and improved design for multiobject iterative auctions. *Man. Sci.* 51, 3 (2005), 419–434.

[20] Jonathan Levin and Andrzej Skrzypacz. 2014. *Are Dynamic Vickrey Auctions Practical?: Properties of the Combinatorial Clock Auction.* Technical Report. Nat. Bureau Econ. Res.

[21] Kevin Leyton-Brown, Mark Pearson, and Yoav Shoham. 2000. Towards a universal test suite for combinatorial auction algorithms. In *Proc. 2nd Conf. Elec. Comm.* ACM, 66–76.

[22] Paul Milgrom. 2000. Putting auction theory to work: The simultaneous ascending auction. *J. Pol. Econ.* 108, 2 (2000).

[23] Debasis Mishra and David C. Parkes. 2007. Ascending price Vickrey auctions for general valuations. *J. Econ. Theory* 132, 1 (2007), 335–366.

[24] Noam Nisan. 2000. Bidding and allocation in combinatorial auctions. In *Proc. 2nd Conf. Elec. Comm.* ACM, 1–12.

[25] Noam Nisan and Ilya Segal. 2006. The communication requirements of efficient allocations and supporting prices. *J. Econ. Theory* 129, 1 (2006), 192–224.

[26] David C. Parkes. 1999. *i*Bundle: An efficient ascending price bundle auction. In *Proc. 1st Conf. Elec. Comm.* 148–157.

[27] David C. Parkes. 2006. Iterative Combinatorial Auctions. In *Combinatorial auctions*, Peter Cramton, Yoav Shoham, and Richard Steinberg (Eds.). MIT Press, Chapter 2.

[28] Alexander Pikovsky, Pasha Shabalin, and Martin Bichler. 2006. Iterative combinatorial auctions with linear prices: Results of numerical experiments. In *8th IEEE Int. Conf. on Enterprise Computation, E-Commerce, and E-Services.*

[29] Tuomas Sandholm. 2006. Optimal winner determination algorithms. In *Combinatorial auctions*, P. Cramton, Y. Shoham, and R. Steinberg (Eds.). MIT Press, Cambridge, MA, 337–368.

[30] T. Sandholm. 2013. Very-Large-Scale Generalized Combinatorial Multi-Attribute Auctions: Lessons from Conducting $60 Billion of Sourcing. In *The Handbook of Market Design*, Nir Vulkan, Alvin E. Roth, and Zvika Neeman (Eds.). Oxford University Press, Chapter 1.

[31] Tobias Scheffel, Alexander Pikovsky, Martin Bichler, and Kemal Guler. 2011. An experimental comparison of linear and nonlinear price combinatorial auctions. *Info. Sys. Res.* 22, 2 (2011), 346–368.

A OMITTED PROOFS

Proof of Proposition 2.1. The argument is standard and we provide a proof for completeness. Let $\mathbf{Y}'$ be an arbitrary feasible allocation. By definition (4), we have that $v_i(Y_i) - p_i(Y_i) \geq v_i(Y_i') - p_i(Y_i') - \epsilon$ for all $i \in N$. Again by definition (4), we have $\sum_{i \in N} p_i(Y_i) \geq \sum_{i \in N} p_i(Y_i')$. Summing up all these inequalities yields $\sum_{i \in N} v_i(Y_i) \geq \sum_{i \in N} v_i(Y_i') - n\epsilon$. □

Proof of Theorem 3.1. For any given round, let $\mathcal{F}'$ denote the set of feasible allocations considered by the auction for revenue-maximization purposes, based on the bids placed so far. With set-packing allocation, the auction selects an allocation from the supply set (3) where $\mathcal{F}$ is replaced by $\mathcal{F}'$. By reasoning analogous to the proof of Proposition 2.1, this implies that the auction terminates with an allocation that maximizes efficiency within the current $\mathcal{F}'$.

Assume that the agents are single-minded, and interested in bundles $X_1, \ldots, X_n$ respectively. Each of these bundles is bid on given zero prices in the first round, and therefore considered for allocation by the auction in all subsequent rounds. In particular, $\mathcal{F}'$ contains an efficient allocation in every round beyond the first, which proves the first claim.

Assume now that agents have general valuations. Let $X_1, \ldots, X_n$ be the bundles bid on by the agents in the first round. As prices are zero in the first round, it is the case that $v_i(X_i) = \max_{Z \in \mathcal{Z}} v_i(Z)$ for all $i \in N$, by straightforward bidding. Let k be the agent for which $v_k(X_k)$ is maximal. Beyond the first round, $\mathcal{F}'$ always contains the allocation where agent k obtains X_k, and the others receive $\emptyset$. The efficiency of this allocation is

$$v_k(X_k) \geq \frac{1}{n} \sum_{i=1}^{n} v_i(X_i) \geq \frac{1}{n} \sum_{i=1}^{n} v_i(Y_i),$$

where $(Y_1, \ldots, Y_n)$ is any efficient allocation among the agents. The first inequality follows from $v_k(X_k) \geq v_i(X_i)$ for all $i \in N$, and the second from $v_i(X_i) \geq v_i(Y_i)$ for all $i \in N$. This proves the second claim. □

B SOLVING THE RESTRICTED PRIMAL

Here we provide the implementation details of solving (RP) via column generation. Recall that it is impractical to enumerate all variables $y(\mathbf{Y})$ for $\mathbf{Y} \in S'(p^t)$ at the outset. We initialize the LP with the single variable $y(\mathbf{Y}^t)$ associated with the provisional allocation. Once solved, we obtain dual variables $\tilde{w}$, $\tilde{\pi}_i$ (for $i \in N$) and $\tilde{\pi}^s$ in analogy with the variables in (D). Let $\tilde{p}$ be the prices defined by coefficients $\tilde{w}$. We then seek a column (i.e., variable) $y(\mathbf{Y})$ with positive *reduced cost*, defined as

$$\text{reduced cost of } y(\mathbf{Y}) = \begin{cases} 1 - \tilde{\pi}^s + \sum_{i \in N} \tilde{p}(Y_i) & \text{if } \mathbf{Y} \in S'(p^t) \\ -\tilde{\pi}^s + \sum_{i \in N} \tilde{p}(Y_i) & \text{otherwise} \end{cases}$$

If there is no such column, then we have solved (RP). Here $-\tilde{\pi}^s$ is just a constant, so generating a column with largest reduced cost amounts to finding an allocation that maximizes revenue with respect to $\tilde{p}$, with a bonus of 1 for allocations in $S'(p^t)$. Recall that we know the optimal revenue with respect to p since we computed $\mathbf{Y}^t$; let r^* be this optimum. To generate a column, we can adapt the IP for revenue-maximization, used to compute the provisional allocation. We introduce another binary variable $z \in \{0,1\}$ and an additional constraint with a sufficiently large constant M.

$$\begin{aligned}
\max_{x,z} \quad & \sum_{i \in N} \sum_{X_i \in \mathcal{B}_i} \tilde{p}^t(X_i)\, x_i(X_i) + z - \tilde{\pi}^s \\
\text{s.t.} \quad & \sum_{i \in N} \sum_{X_i \in \mathcal{B}_i,\, a \ni X_i} x_i(X_i) \leq 1 \qquad (a \in M) \\
& \sum_{X_i \in \mathcal{B}_i} x_i(X_i) \leq 1 \qquad (i \in N) \\
& r^* - \sum_{i \in N} \sum_{X_i \in \mathcal{B}_i} p^t(X_i)\, x_i(X_i) \leq M(1 - z)
\end{aligned}$$

If the objective value of this IP is non-positive, we are done—there is no column with positive reduced cost. Otherwise we can add column $y(\mathrm{Y})$ to the (RP) formulation where Y is the allocation defined by the optimal $x_i(X_i)$ variables.